January 23, 2025

Announcements

- Talk topic / SSH key
- cal / Office hours
- HW1 next week

Goals

- Intro cont'd
- "Armchair architecture"

Review

- Abstractions
- Models

Approaches to High Performance

- Libraries (seen)
- Black-box Optimizing Compilers
- Compilers with Directives
- Code Transform Systems
- "Active Libraries"

"established"

Q: Give examples of the latter two.

Libraries: A Case Study

$$(C_{ij})_{i,j=1}^{m,n} = \sum_{k=1}^{\ell} A_{ik} B_{kj}$$

Demo: intro/DGEMM Performance

-
-

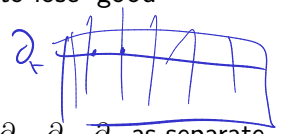
Do Libraries Stand a Chance? (in general)

- Tremendously successful approach — Name some examples

BLAS, LAPACK, OpenCV, Deal.ii, MFEM,

- Saw: Three simple integer parameters suffice to lose 'good' performance
  - Recent efforts: e.g. Batch BLAS
- Separation of Concerns

  Example: Finite differences – e.g. implement ∂_x, ∂_y, ∂_z as separate (library) subroutines — What is the problem?



Redundant data access / fusion

- Flexibility and composition

(Black-Box) Optimizing Compiler: Challenges

Why is black-box optimizing compilation so difficult?

- Application developer knowledge lost
  - Simple example: "Rough" matrix sizes
  - Data-dependent control flow
  - Data-dependent access patterns
  - Activities of other, possibly concurrent parts of the program
  - Profile-guided optimization can recover some knowledge
- Obtain proofs of required properties
- Size of the search space
  Consider http://polaris.cs.uiuc.edu/publications/padua.pdf

Directive-Based Compiler: Challenges

What is a directive-based compiler?

- Generally same as optimizing compiler
- Make use of extra promises made by the user
- What should the user promise?
- Ideally: feedback cycle between compiler and user
  - Often broken in both directions
  - User may not know what the compiler did
  - Compiler may not be able to express what it needs
- Directives: generally not mandatory

prescriptive / descriptive

Lies, Lies Everywhere

- Semantics form a contract between programmer and language/environment
- Within those bounds, implementation has full freedom
- True at every level:
  - Assembly
  - "High-level" language (C)

  Give examples of lies at these levels:

for (i)
{
a[i+4] = 10
}

Instruction reordering
"Strength reduction"

One approach: *Lie to yourself*

- "Domain-specific languages" ← A fresh language, I can do what I want!
- Consistent semantics are notoriously hard to develop
  - Especially as soon as you start allowing subsets of even (e.g.) C's integers

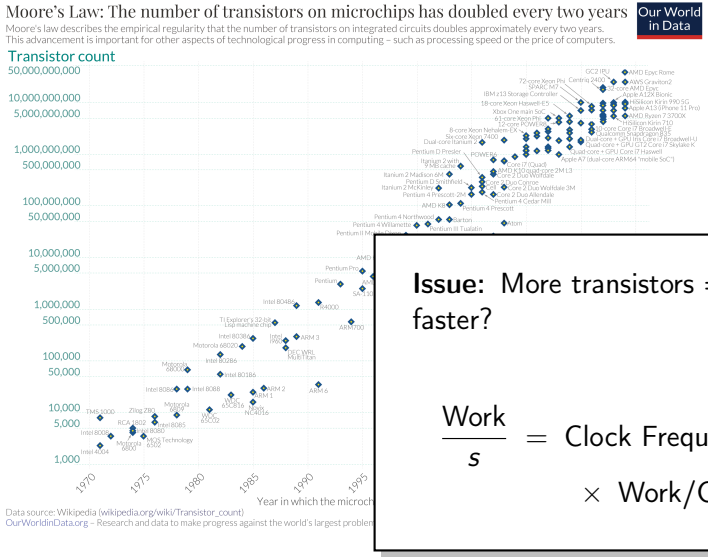
Class Outline

High-level Sections:

- Intro, Armchair-level Computer Architecture
- Machine Abstractions
- Performance: Expectation, Experiment, Observation
- Programming Languages for Performance
- Program Representation and Optimization Strategies
- Code Generation/JIT

Outline

Moore's Law



Issue: More transistors = faster?

$$\frac{\text{Work}}{s} = \text{Clock Frequency} \times \text{Work/Clock}$$

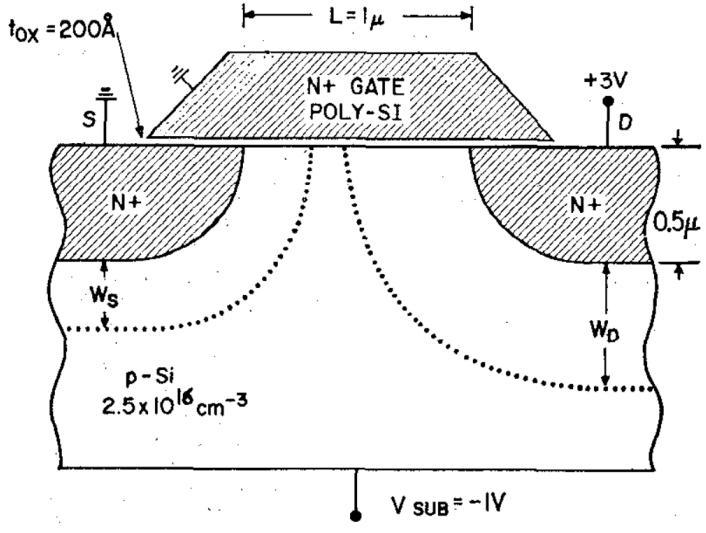
Dennard Scaling of MOSFETs

Parameter	Factor
Dimension	$1/\kappa$
Voltage	$1/\kappa$
Current	$1/\kappa$
Capacitance	$1/\kappa$
Delay Time	$1/\kappa$
Power dissipation/circuit	$1/\kappa^2$
Power density	1

[Dennard et al. '74, via Bohr '07]

- Frequency = Delay time^{-1}

MOSFETs ("CMOS" – "complementary" MOS): Schematic



[Dennard et al. '74]